

Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1-6. (Canceled).

7. (Currently Amended) A method of data processing using a processor comprising a reconfigurable field of data processing cells and a register, wherein the register has a data stream memory operated as a First-In-First-Out (FIFO) memory to store at least one data vector, the method comprising:

providing a program corresponding to a sequence of compilable high-level language instructions;

determining, for the reconfigurable field of data processing cells, a set of configurations of the reconfigurable field of data processing cells, with respect to at least one of a function and an interconnection of the reconfigurable field of data processing cells, with execution of which configurations the program is run;

determining, for each of the configurations, a respective maximum allowed execution runtime prior to lapse of which the respective configuration is uninterruptible, and in response to lapse of which the reconfigurable field of data processing cells is reconfigured with a different configuration;

executing the configurations; and

during the executing:

storing, in the data stream memory, at least one of the data stream and parts of the data stream; and

for each configuration, monitoring the respective maximum allowed execution runtime in order to interrupt processing with the configuration if the respective maximum allowed execution runtime is exceeded.

8. (Previously Presented) The method as recited in claim 7, further comprising:

using at least one of: i) a register allocation device to allocate the register, and ii) a register releasing device to release the register.

U.S. Patent App. Ser. No. 10/501,845
Attorney Docket No. 02885/86
Reply to Final Office Action

9. (Previously Presented) The method as recited in claim 8, wherein the register allocation device is preserved over multiple reconfigurations of the reconfigurable field of data processing cells.

10. (Previously Presented) The method as recited in claim 7, wherein the register is a Random Access Memory (RAM) cell.

11. (Previously Presented) The method as recited in claim 7, wherein the program includes a multitask application, the method further comprising:

using the register:

to provide read and write access when a virtual FIFO dividing line is implemented; and

for execution of at least one of two different tasks of the multitask application.

12. (Previously Presented) The method as recited in claim 7, further comprising:
using at least one memory unit as a stack and to indicate at least one of a stack underflow state and a stack overflow state.

13. (Previously Presented) The method as recited in claim 12, wherein the at least one of the underflow state and overflow state is of an operating system unit.

14. (Canceled).

15. (Previously Presented) The method as recited in claim 7, wherein a watchdog is used to recognize an exceedance of each respective maximum allowed execution runtime.

16. (Previously Presented) The method as recited in claim 15, wherein any one of the configurations that exceeds its respective maximum allowed execution runtime is treated as illegal.

17. (Previously Presented) The method as recited in claim 7, wherein any one of the configurations that exceeds its respective maximum allowed execution runtime is treated as illegal.

18. (Canceled).

19. (Previously Presented) The method as recited in claim 7, wherein an operating system performs a predefined step in response to an exceedance by a configuration of the configuration's maximum allowed execution runtime.

20. (Previously Presented) The method as recited in claim 7, wherein at least one of the configurations calls another of the configurations as a sub-routine.

21. (Previously Presented) The method as recited in claim 15, wherein a signal of the watchdog initiates a system trap.

22. (Previously Presented) The method as recited in claim 21, wherein, in response to the system trap, an operating system performs steps defined for a response to an invalid instruction.

23. (New) A method of data processing using a processor comprising a reconfigurable field of data processing cells and a register, wherein the register has a data stream memory operated as a First-In-First-Out (FIFO) memory to store at least one data vector, the method comprising:

providing a program corresponding to a sequence of compilable high-level language instructions;

determining, for the reconfigurable field of data processing cells, a set of configurations of the reconfigurable field of data processing cells, with respect to at least one of a function and an interconnection of the reconfigurable field of data processing cells, wherein an instruction of the program is executed using a plurality of the configurations;

determining, for each of the configurations, a respective maximum allowed execution runtime prior to lapse of which the respective configuration is uninterruptible;

implementing the configurations; and

during the implementing:

storing, in the data stream memory, at least one of the data stream and parts of the data stream; and

for each configuration, monitoring the respective maximum allowed execution runtime in order to interrupt processing with the configuration if the respective maximum allowed execution runtime is exceeded.

24. (New) A method of data processing using a processor comprising a reconfigurable field of data processing cells and a register, wherein the register has a data stream memory operated as a First-In-First-Out (FIFO) memory to store at least one data vector, the method comprising:

providing a program corresponding to a sequence of compilable high-level language instructions;

determining, for the reconfigurable field of data processing cells, a set of configurations of the reconfigurable field of data processing cells, with respect to at least one of a function and an interconnection of the reconfigurable field of data processing cells, with execution of which configurations the program is run, wherein, for each of at least one of the configurations, a plurality of instructions of the program are executable via single instance of the respective configuration;

determining, for each of the configurations, a respective maximum allowed execution runtime prior to lapse of which the respective configuration is uninterruptible;

executing the configurations; and

during the executing:

storing, in the data stream memory, at least one of the data stream and parts of the data stream; and

for each configuration, monitoring the respective maximum allowed execution runtime in order to interrupt processing with the configuration if the respective maximum allowed execution runtime is exceeded.